Spring Boot实际应用讲解（三）： 表单验证


ZYRzyr (/u/f8ff63b17fc7) + 关注
2017.11.14 16:48* 字数 559 阅读 377 评论 0 喜欢 3
(/u/f8ff63b17fc7)

> 文/ZYRzyr (https://www.jianshu.com/u/f8ff63b17fc7)
> 原文链接:http://www.jianshu.com/p/a2b4e61b5532
> (https://www.jianshu.com/p/a2b4e61b5532)

本文提纲

一、表单验证实例
二、验证注解说明
三、最后

本文运行环境

```
Ubuntu 16.04 LTS
JDK 8 +
IntelliJ IDEA ULTIMATE 2017.2
Maven 3.5.0
Spring Boot 1.5.8.RELEASE
```

一、表单验证实例

表单验证，即校验用户提交的数据的合理性，如是否为空，数据长度，数据类型等等，好的验证，能防止垃圾数据的产生。

1.1 新建类实体类

```
package com.zyr.demo.domain;

import javax.validation.constraints.Min;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;

public class User {

    @NotNull(message = "用户名不能为空")
    @Size(min = 1, message = "用户名不能为空") 字符串必须有此注解，否则无法验证传的值""
    private String name;

    @NotNull(message = "年龄不能为空")
    @Min(value = 18, message = "必须是成年人")
    private Integer age;

    @NotNull(message = "密码不能为空")
    @Size(min = 6, max = 15, message = "密码长度为6–15位")
    private String password;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Integer getAge() {
        return age;
    }

    public void setAge(Integer age) {
        this.age = age;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    @Override
    public String toString() {
        return "User{" +
                "name='" + name + '\'' +
                ", age=" + age +
                ", password='" + password + '\'' +
                '}';
    }
}
```

1.2 新建Controller

```
package com.zyr.demo.controller;

import com.zyr.demo.domain.User;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RestController;

import javax.validation.Valid;

@RestController
public class UserController {

    @PostMapping("/signUp")  //POST请求方式
    public String signUp(@Valid User user, BindingResult bindingResult) {
        System.out.println("name=" + user.getName());
        System.out.println("age=" + user.getAge());
        System.out.println("password=" + user.getPassword());

        if (bindingResult.hasErrors()) {
            return bindingResult.getFieldError().getDefaultMessage();
        }

        return "success";
    }
}
```

1. 对 `signUp` 的 `POST` 请求参数即为 `User` 类中的字段名： `name` ， `age` ， `password` 。 `Spring Boot` 会将其自动组合成 `User` 实例；
2. `@Valid` ： 表示该参数需要进行验证，验证点即为上面 `User` 类中的字段上的各注解；
3. `BindingResult` ： 此参数包含错误信息，即 `User` 字段注解中的 `message` ；
4. `signUp` 方法打印请求参数的值，若未通过验证则返回错误信息，错误信息即 `User` 类字段注解中的 `message` ， 若验证成功直接返回 `success` 。

1.3 测试

新建测试类 `UserControllerTest` (此处未写完所有测试用例)：

```
package com.zyr.demo.controller;

import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.test.context.junit4.SpringRunner;
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;
import org.springframework.test.web.servlet.result.MockMvcResultMatchers;

@RunWith(SpringRunner.class)
@SpringBootTest
@AutoConfigureMockMvc
public class UserControllerTest {

    @Autowired
    private MockMvc mockMvc;

    @Test
    public void testSignUp_success() throws Exception {
        mockMvc.perform(MockMvcRequestBuilders.post("/signUp")
                .param("name", "Bob")
                .param("age", "20")
                .param("password", "123456"))
                .andExpect(MockMvcResultMatchers.status().isOk())
                .andExpect(MockMvcResultMatchers.content().string("success"));
    }

    @Test
    public void testSignUp_name_null() throws Exception {
        mockMvc.perform(MockMvcRequestBuilders.post("/signUp")
                .param("name", "")
                .param("age", "20")
                .param("password", "123456"))
                .andExpect(MockMvcResultMatchers.status().isOk())
                .andExpect(MockMvcResultMatchers.content().string("用户名不能为空"));
    }

    @Test
    public void testSignUp_age_error() throws Exception {
        mockMvc.perform(MockMvcRequestBuilders.post("/signUp")
                .param("name", "Bob")
                .param("age", "10")
                .param("password", "123456"))
                .andExpect(MockMvcResultMatchers.status().isOk())
                .andExpect(MockMvcResultMatchers.content().string("必须是成年人"));
    }
}
```

运行测试用例，全部成功，说明正确。

二、验证注解说明

除了上面 `User` 类使用的注解，还有其它很多类型的注解：

注解	作用
`@AssertFalse`	验证 `boolean` 对象是否是 `false`
`@AssertTrue`	验证 `boolean` 对象是否是 `true`
`@DecimalMax`	验证 `Number` 和 `String` 对象是否小等于指定的值，小数存在精度
`@DecimalMin`	验证 `Number` 和 `String` 对象是否大等于指定的值，小数存在精度
`@Digits`	验证 `Number` 和 `String` 的构成是否合法
`@Past`	验证 `Date` 和 `Calendar` 对象是否在当前时间之前
`@Future`	验证 `Date` 和 `Calendar` 对象是否在当前时间之后
`@Max`	验证 `Number` 和 `String` 对象是否小等于指定的值

注解	作用
`@Min`	验证 `Number` 和 `String` 对象是否小等于指定的值
`@NotNull`	验证对象不为空
`@Null`	验证对象为空
`@Pattern`	验证 `String` 对象是否符合正则表达式的规则
`@Size`	验证对象（`Array`,`Collection`,`Map`,`String`）长度是否在给定的范围之内

三、最后

本文介绍了 `Spring Boot` 是如何使用注解，就能方便的进行表单验证的。

本文代码已上传至我的GitHub仓库 (https://link.jianshu.com?t=https://github.com/ZYRzyr/SpringBootDemo)，进入以后将branches (https://link.jianshu.com?t=https://github.com/ZYRzyr/SpringBootDemo/branches)切换为3-FormValidate (https://link.jianshu.com?t=https://github.com/ZYRzyr/SpringBootDemo/tree/3-FormValidate)即可看见。

前篇：
Spring Boot实际应用讲解（一）： Hello World (https://www.jianshu.com/p/60f7e025c680)
Spring Boot实际应用讲解（二）： 配置详解 (https://www.jianshu.com/p/d4c7f33c9b37)

后续将推出以下文章，敬请关注！

Spring Boot实际应用讲解（四）： RESTful API (https://www.jianshu.com/p/e907595e9d1d)
Spring Boot实际应用讲解（五）： AOP之请求日志 (https://www.jianshu.com/p/93216bf41182)
Spring Boot实际应用讲解（六）： MySQL + Spring-data-jpa(Hibernate) (https://www.jianshu.com/p/b204472d8126)
Spring Boot实际应用讲解（七）： 统一异常处理
Spring Boot实际应用讲解（八）： MySQL + Mybatis
Spring Boot实际应用讲解（九）： MySQL + Mybatis + Redis

文中若有错之处，还请各位批评指正，谢谢！

> 原文作者/ZYRzyr (https://www.jianshu.com/u/f8ff63b17fc7)
> 原文链接:http://www.jianshu.com/p/a2b4e61b5532 (https://www.jianshu.com/p/a2b4e61b5532)